

## A Stabilizing AQM Based on Virtual Queue Dynamics in Supporting TCP with Arbitrary Delays

Ki Baek Kim, Ao Tang, Steven H. Low

Departments of CS and EE  
California Institute of Technology, Pasadena, CA 91125  
kbb@di.ens.fr, aotang@caltech.edu, slow@caltech.edu

**Abstract**— This paper studies how to design a stabilizing AQM in supporting TCP (Transmission Control Protocol) with arbitrary delays. For the well-known AIMD (Additive Increase Multiplicative Decrease) dynamic model of TCP, our study shows that we can compensate for arbitrary delays explicitly by applying a modified virtual queue dynamics that makes the equilibrium queuing delay zero. This study also verifies that a simplified AQM AVQ (Adaptive Virtual Queue) is the state-feedback control for the AIMD model based on the virtual queue dynamics.

### I. INTRODUCTION

Since TCP Reno/AQM Droptail has been proposed in [1], the current internet is still using Reno/Droptail and its variants as a congestion control strategy.

Droptail, which drops packets when the queue is full, is not appropriate as a feedback control for high-speed networks since links are likely to have large queues in high-speed networks and this may increase the average delay in the network. More importantly, Droptail can often cause the global synchronization [2] and thus have low throughput. In order to overcome these problems, RED (Random Early Detection) has been suggested in [3] which randomly drops packets proportionally to the average queue length. Since then, there have been a lot of investigations about how to tune design parameters in RED [4], [5]. They show that RED is not enough to stabilize TCP and thus not easy to fully utilize the given network resources. As a result, new AQM algorithms such as BLUE [6], AVQ (Adaptive Virtual Queue) [7] and REM (Random Exponential Marking) [8] have been suggested. However, all these papers did not address appropriately how to tune gains of their AQM structures for closed-loop stability since they lack of a TCP dynamic model, in spite that stabilizing the closed-loop system is necessary to get the maximum utilization.

In order to address this problem, paper [9] has developed a dynamic model to reflect AIMD (Additive Increase Multiplicative Decrease) mode of TCP and paper [10] has applied the transfer function approach to the stabilizing RED design problem based on the AIMD model. As a sequence, papers [11], [12] have investigated how to scale gains of PI-type REM (Proportional-Integral) in terms of queue length and

P-type AVQ in terms of aggregate, respectively, for closed-loop stability. The papers [13], [14] have suggested PI-type AQM in terms of aggregate in an inner loop. However, all these results based on the transfer function approach do not consider *what is a natural state-feedback control* to stabilize the given TCP and queue dynamics. In addition, they did not consider how to compensate for delays explicitly even if they know the previous dynamic information and delays, where this kind of control strategy is called delay-independent (or memoryless) control in the control literature. It is well-known that the delay-independent control has a limit on performance in the presence of a large delay [15], [16] compared with the delay-dependent (or memory) control, i.e., we cannot fully utilize the given network resources.

Recently, the papers [17], [18], [19] formulate a simplified version of the AQM design problem for the well-known AIMD dynamic model in [9] as state-space models. Thereby, they showed that PD-type (Proportional-Derivative) AQM is a natural state-feedback control structure to stabilize TCP Reno for the first time in the networking literature to our knowledge, where RED can be classified as P-type AQM in terms of queue length. They also proposed delay-dependent AQMs to compensate for delays in congestion measure explicitly and discussed the impact of different AQM structures on performance. However, they assume that the forward delay between source and link is zero when they compensate for delays explicitly. Through *ns* simulations, they show that this assumption cannot be ignored in the presence of large delays. Thus, it is necessary to investigate how to design a stabilizing AQM without any assumption about delay.

This paper attempts to address this issue by applying a modified virtual queue dynamics. For simplicity of designing a stabilizing AQM as in [10], [11], [13], [14], we assume single link and homogeneous sources with the same window, and no short flows, where the real network is asymmetric as shown in [20]. We also do not consider input/state constraints and uncertainty which are intrinsically included in real networks and *ns* simulator, although our work can be extended to more realistic problems using modern control theories and technologies.

Then, our study shows that we can compensate for arbitrary delays explicitly by employing a modified virtual queue dynamics that makes the equilibrium queuing delay zero. As a subsidiary result of this study, we verify that a

The first author moves to INRIA-ENS, ENS-DI 45 rue d'Ulm 75005, Paris FRANCE.

simplified AVQ, which is P-type delay-independent AQM in terms of aggregate, is a state-feedback control for the AIMD model based on the virtual queue dynamics. Finally, we illustrate the proposed results via the non-deterministic nonlinear *ns* simulator. The simulation results should be considered as backing up the proposed theoretical results qualitatively rather than as their direct illustration.

## II. AQM DESIGN COMPENSATING ARBITRARY DELAYS

Consider  $N$  homogeneous TCP Reno sources with the same window size modeled by [9]

$$w_s(t) = \frac{w_s(t - \tau_s(t))}{d_s + b(t - \tau_s(t))/c} (1 - p(t - \tau_s^b(t))) \frac{1}{w_s(t)} - \frac{w_s(t - \tau_s(t))}{d_s + b(t - \tau_s(t))/c} p(t - \tau_s^b(t)) \frac{w_s(t)}{2} \quad (1)$$

sharing a single link of capacity  $c$  with the queue dynamics modeled by

$$\dot{b}(t) = -c + \sum_{s=1}^N \frac{w_s(t - \tau_s^f(t))}{d_s + b(t - \tau_s^f(t))/c} = -c + y(t),$$

where  $w_s(t)$  is TCP window size in packets of source  $s$  at time  $t$ ,  $\tau_s^f(t)$  is the forward delay from source to link,  $\tau_s^b(t)$  is the backward delay from link to source,  $\tau_s(t)$  is the round trip time ( $\tau_s(t) = \tau_s^f(t) + \tau_s^b(t)$ ),  $p(t)$  is the loss probability at time  $t$  that is implemented at links,  $b(t)$  is the real queue length,  $y(t)$  is aggregate rate, and  $c$  is the link capacity in packets/sec. We define the source rate by  $x_s(t) = w_s(t)/(d_s + b(t)/c)$ , where  $d_s$  is the round trip propagation delay of source  $s$ . As in [10], we assume sources are identical  $d_s = d$  and all have the common window  $w_s(t) \equiv w(t)$ ; we assume delays take their equilibrium values and are constant, so that  $\tau_s = \tau$ .

In the above model, the first and second parts of the first equation represent AI and MD behaviors at sources, respectively, where the second equation represents the real queue dynamics at link. For this AIMD model, our papers [17], [18], [19] propose what is a natural feedback control structure for stabilizing the given TCP, how to compensate for delays explicitly, and a unified AQM framework based which interprets a simplified RED [3] and REM/PI (Proportional-Integral) [8], [11] as different approximations. However, they assume that the forward delay  $\tau_s^f(t)$  is zero since the nonzero  $\tau_s^f(t)$  produces a state-delay for the linearized system of the AIMD model (1) that makes it impossible to compensate for delays in a closed-form. Through *ns* simulations, they show that this assumption cannot be ignored in the presence of large delays. This problem naturally leads to the present work.

The key idea to overcome this problem is to add the following modified virtual queue dynamics ( $\dot{b}(t)$ ) at the

router<sup>1</sup> to the AIMD model (1):

$$\begin{aligned} \dot{\tilde{b}}(t) &= \gamma_1 \left[ -\gamma_2 c + \sum_{s=1}^N \frac{w_s(t - \tau_s^f(t))}{\tau_s(t - \tau_s^f(t))} \right] \\ &= \gamma_1 [-\gamma_2 c + y(t)], \end{aligned}$$

where  $\tilde{b}(t)$  is the virtual queue length, and  $\gamma_1 (> 0)$  and  $\gamma_2$  are free design parameters. In this paper, we assume that  $0 < \gamma_2 < 1$ .

Let  $(w^*, b^*, p^*)$  be the equilibrium point. If  $0 < \gamma_2 < 1$ , the equilibrium point of  $y(\cdot)$  is  $y^* = \gamma_2 c$  due to the virtual queue dynamics. Hence, we have  $b^* = 0$ ,  $\tau^* = \tau^f + \tau^b = d$ ,  $w^* = \frac{d}{N} \gamma_2 c$ , and  $p^* = \frac{2}{2 + w^*} = \frac{2N^2}{2N^2 + (d\gamma_2 c)^2}$ .

Since the virtual queue dynamics is dominant near the equilibrium point, we approximate that the real queue  $b(\cdot)$  is zero near the operating point from now on in this paper.

Then, system (1) based on the above virtual queue dynamics is converted to the equivalent form

$$\begin{aligned} \ddot{\tilde{b}}(t) &= \frac{\gamma_1 N (\dot{\tilde{b}}(t - d) + \gamma_1 \gamma_2 c)}{d^2 (\dot{\tilde{b}}(t) + \gamma_1 \gamma_2 c)} - \\ &\quad \left\{ \frac{\gamma_1 N (\dot{\tilde{b}}(t - d) + \gamma_1 \gamma_2 c)}{d^2 (\dot{\tilde{b}}(t) + \gamma_1 \gamma_2 c)} + \right. \\ &\quad \left. \frac{(\dot{\tilde{b}}(t - d) + \gamma_1 \gamma_2 c)(\dot{\tilde{b}}(t) + \gamma_1 \gamma_2 c)}{2\gamma_1 N} \right\} p(t - d)(2) \\ &= f(\dot{\tilde{b}}(t), \dot{\tilde{b}}(t - d), p(t - d)). \end{aligned}$$

In general, it is difficult to systematically find a nonlinear function  $p(t)$  which guarantees the global asymptotic stability for nonlinear dynamical systems with delays. As a starting point to address these problems and as a method to avoid the nonnegative constraint, the present paper considers an equilibrium point with positive values and studies the linearized version of the derived dynamical system (2) on  $\dot{\tilde{b}}(t)$ ,  $\dot{\tilde{b}}(t - d)$  and  $p(t - d)$  near the equilibrium point.

From (2), we can get the following model of the linearized TCP and virtual queue dynamics:

$$\delta \ddot{\tilde{b}}(t) = A_2 \delta \dot{\tilde{b}}(t) + B_1 \delta p(t - d), \quad (3)$$

where  $\delta \ddot{\tilde{b}}(t) = \ddot{\tilde{b}}(t)$ ,  $\delta \dot{\tilde{b}}(t) = \dot{\tilde{b}}(t)$ ,  $\delta p(t) = p(t) - p^*$ ,  $\delta \dot{\tilde{b}}(0)$  and  $\{\delta p(\sigma), \sigma \in [-d, 0]\}$  are given,  $A_2 = -\frac{2\gamma_2 c N}{2N^2 + \gamma_2^2 c^2 d^2}$  and  $B_1 = -\frac{\gamma_1 (2N^2 + \gamma_2^2 c^2 d^2)}{2d^2 N}$ . Refer to Section VI in Appendix for derivation of (3).

Note that the above model *does not include any time delay in the state variable  $\delta \dot{\tilde{b}}(t)$  and only includes the time delay in the control  $\delta p(t - d)$* . This interesting and non-intuitive feature allows us to employ the explicit compensation method used in our companion papers [17], [18], [19] as shown in the following subsection.

<sup>1</sup>If  $\gamma_1 = \gamma_2 = 1$ , the virtual queue dynamics is equal to the real queue dynamics, and the AVQ in [12] assumes  $\gamma_1 < 0$  while we assume  $\gamma_1 > 0$  in this paper.

From the above model, we can naturally get a P-type state-feedback AQM in terms of aggregate:

$$\delta p(t) = H_P \delta \dot{b}(t) \quad (4)$$

if we ignore the time delay  $d$  (i.e.,  $d = 0$ ) in the control  $\delta p(t)$ . How to compensate for the delay explicitly and how to obtain a stabilizing gain  $H_P$  is discussed later in this paper.

The derived P-type AQM structure, which we also did not expect, is interesting again as follows: If we linearize the AVQ  $p(t) = p(\dot{b}(t))$  in [7], [12], we also have a P-type control structure  $\delta p(t) = H_P^p \delta \dot{b}(t)$ , i.e., our study supports their argument that the AVQ is an appropriate control structure to stabilize TCP Reno based on the virtual queue dynamics in the absence of delays

As shown in [8], [11], [13], [17], [18], [19], we can make the steady-state tracking error approach zero faster by adding integral control action [21] to the original system as follows:

$$\dot{z}(t) = Az(t) + B\delta p(t-d), \quad (5)$$

where  $z(0)$  and  $\{\delta p(\sigma), \sigma \in [-d, 0]\}$  are given,  $z(t) = \begin{bmatrix} \delta \dot{b}(t) \\ \delta \ddot{b}(t) \end{bmatrix}$ ,  $A = \begin{bmatrix} 0 & 1 \\ 0 & A_2 \end{bmatrix}$ ,  $B = \begin{bmatrix} 0 \\ B_1 \end{bmatrix}$ . Refer to Section VI in Appendix for derivation of (5). For the above minimal representation with state variables  $(\delta \dot{b}(t), \delta \ddot{b}(t))$ , we can get a PI-type state-feedback AQM

$$\delta p(t) = H z(t) = H_I \delta \dot{b}(t) + H_P \delta \ddot{b}(t) \quad (6)$$

if we ignore the time delay  $d$  (i.e.,  $d = 0$ ) in the control  $\delta p(t)$ .

Note that the proposed PI-type AQM does not require the equilibrium point  $p^*$  but requires the initial loss probability  $p(t_0)$ , while the proposed P-type AQM needs  $p^*$ .

Now, we are ready to show how to compensate the delays and how to get a stabilizing gain. Although the following results can easily be obtained from [17], [18], [19], we introduce them in order for this paper to be self-contained. In the next section, we consider only the PI-type AQM structure (6) since we can handle P-type AQM (4) as a special case of the PI-type AQM.

### III. EXPLICIT DELAY COMPENSATION AND A STABILIZING GAIN

The delayed system (5) can be converted to the equivalent nominal system

$$\dot{s}_e(t) = As_e(t) + \hat{B}\delta p(t), \quad (7)$$

where  $s_e(t) = [s_1(t), s_2(t)]^T$ ,  $\hat{B} = [0, \hat{B}_1]^T$ ,  $\hat{B}_1 = \frac{B_1 A_2 e^{-A_2 d}}{(1 - e^{-A_2 d})}$ ,

$$s_1(t) = \frac{A_2 e^{-A_2 d}}{(1 - e^{-A_2 d})} (\delta \dot{b}(t) + \dot{u}_{1d}(t)) + \delta \ddot{b}(t) + \dot{u}_{2d}(t)$$

$$s_2(t) = \frac{A_2}{(1 - e^{-A_2 d})} (\delta \ddot{b}(t) + \dot{u}_{2d}(t))$$

$$\begin{bmatrix} \dot{u}_{1d}(t) \\ \dot{u}_{2d}(t) \end{bmatrix} = -\frac{B_1}{A_2} \int_{-d}^0 \begin{bmatrix} 1 - e^{-(\sigma+d)A_2} \\ -A_2 e^{-(\sigma+d)A_2} \end{bmatrix} \delta p(\sigma+t) d\sigma.$$

Refer to Section VII in Appendix for derivation of (7). It is easy to see that the closed-loop system of (7) is asymptotically stable if and only if the original system (5) is asymptotically stable.

From the above state-space model, we can naturally get PI-type delay-dependent (or memory) AQM

$$\delta p(t) = H_I^d s_1(t) + H_P^d s_2(t). \quad (8)$$

Now, we get a stabilizing optimal gain of the proposed control structures. For this purpose, we consider the following performance index:

$$\min_{\delta p(\cdot)} \int_t^{t+\infty} (Q_1 s_1^2(\sigma) + Q_2 s_2^2(\sigma) + \delta p^2(\sigma)) d\sigma, \quad (9)$$

where  $Q_1 > 0$  and  $Q_2 \geq 0$ .

**Proposition 1:** The stabilizing optimal gain of the proposed PD-type AQM (8), which minimizes the transient cost (9), is given by

$$H_I^d = \sqrt{Q_1}, \quad H_P^d = -\frac{A_2 + \sqrt{A_2^2 + \hat{B}_1^2 Q_2 + 2\sqrt{\hat{B}_1^2 Q_1}}}{\hat{B}_1}.$$

**Proof:** The optimal control that minimizes (9) is given by

$$\delta p^*(t) = -\hat{B}^T K_e s_e(t).$$

□

Proposition 1 implies that the solution of the problem (9) is a stabilizing AQM algorithm, specified by  $(H_I^d, H_P^d)$ . Conversely, given any AQM of this structure, it solves (9) with appropriate weights  $Q_i$ , as the next result says. It can be easily proved from Proposition 1.

**Proposition 2:** Given a stabilizing AQM  $\delta p(t) = [H_1, H_2] s_e(t)$  that satisfies  $\hat{B}_1 H_1 < 0$  and  $A_2 + \hat{B}_1 H_2 < 0$ , it solves problem (9) with weights:

$$Q_1 = H_1^2, \quad Q_2 = \frac{H_2^2 \hat{B}_1 + 2H_2 A_2 + 2H_1}{\hat{B}_1}. \quad (10)$$

**Proposition 3:** Given the eigenvalues  $\lambda_1$  and  $\lambda_2$  of the closed-loop system (7), where real parts of  $\lambda_1$  and  $\lambda_2$  are negative, it solves the problem (9) with weights:

$$Q_1 = \frac{(\lambda_1 \lambda_2)^2}{\hat{B}_1^2}, \quad Q_2 = \frac{\lambda_1^2 + \lambda_2^2 - A_2^2}{\hat{B}_1^2}. \quad (11)$$

In order for an AQM  $\delta p(t) = [H_1, H_2] s_e(t)$  to be a stabilizing optimal control when the system is controllable and observable for system (5), it should satisfy  $H_1 > 0$ . The implication of  $H_1 = 0$  is that at least one of the eigenvalues  $\lambda_i$  is zero, implying that the convergence rate is very slow and the original nonlinear system could be unstable according to the center manifold theorem [22]. As shown in Proposition 2, if  $H_2 = 0$ , then we have  $\lambda_1 + \lambda_2 = A_2 + \hat{B}_1 H_2 \leq A_2$  where the last inequality follows from the fact that  $A_2 < 0$ ,  $\hat{B}_1 < 0$

and  $H_2 \geq 0$ . Since all eigenvalues should have negative real parts for the closed-loop stability, the above inequality means that the sum of the real parts of the eigenvalues is less negative when  $H_2 = 0$  than when  $H_2 > 0$ . This suggests that we need to add  $P$ -type control in order to make the dynamics move faster.

#### IV. ns SIMULATIONS

In this section, we illustrate performance of the derived structures via two simulation examples. Note that the proposed AQMs, which are designed for the linearized system (3), are implemented in *ns* simulator which has stochastic and nonlinear dynamics. Thus, the simulation results should be considered as backing up our study qualitatively rather than its direct illustration.

For ease of selection of stabilizing optimal gains, we set two eigenvalues of the closed-loop system to be equal (i.e.,  $\lambda = \lambda_1 = \lambda_2$ ). Then, we have only to design  $\lambda < 0$  in order to get the stabilizing optimal gains. The dynamic behavior of the closed-loop system depends on how to design  $\lambda$ . A larger  $|\lambda|$  makes the system move faster, i.e., makes the system go to the equilibrium faster. However, if the value of  $|\lambda|$  is too high, the closed-loop system can also easily deviate from the equilibrium point due to nonlinearity and randomness of *ns* simulator. Since the stabilizing gains in this paper are derived from the deterministic linearized model near the equilibrium point, the value of  $|\lambda|$  should not be so high.

Similarly to [10], [11], we simulate a single bottleneck link with capacity  $c = 4000$  pkts/sec shared by  $N = 100$  Reno sources. The AQM at the bottleneck link uses ECN marking. If there is no specification, we assume that the propagation delay is  $d_s = 250$ ms. Marking probability of each AQM is updated every 2ms, i.e., the sampling time is  $T_s = 2$ ms ( $\tilde{b}(t) = \tilde{b}(t - T_s) + T_s \gamma_1 (y(t - T_s) - \gamma_2 c)$ ). The simulation duration is 30 sec, where this duration is enough to illustrate the proposed results. For all simulations, we set  $\gamma_1 = 1$ ,  $\gamma_2 = 0.95$ , and  $\lambda = -1.35$ , where the maximum throughput is proportional to  $\gamma_2 \in (0, 1)$ . Thus,  $\gamma_2$  should be enough large. If  $d_s$  is too small, the equilibrium loss probability  $p^*$  becomes high and thus TCP Reno can experience time-out many times. Since the TCP Reno model (1) works only when  $p^*$  is small, we do not consider the case that  $d_s$  is small.

##### A. P-type and PI-type memoryless AQMs

In this example, we illustrate performances of the proposed P-type and PI-type memoryless AQMs.

P and PI-type memoryless AQMs are implemented by

$$p(t) = p^* - \frac{(A_2 - \lambda)}{B_1} \gamma_1 (y(t) - \gamma_2 c)$$

$$p(t) = p(t_0) + H_1(\tilde{b}(t) - \tilde{b}(t_0)) + \gamma_1 H_2(y(t) - y(t_0)),$$

respectively, where  $H_1 = -\frac{\lambda^2}{B_1}$ , and  $H_2 = -\frac{(A_2 - 2\lambda)}{B_1}$ .

Figure 1 illustrates that PI-type memoryless AQM (92 %) seems to have a higher utilization than P-type memoryless

AQM (77 %), while both of them have a very small queuing delay and are stabilizing.

##### B. PI-type memoryless and PI-type memory AQMs

In this example, we illustrate the performance of a memory control structure.

PI-type memory AQM is implemented by

$$p(t) = p(t_0) + H_1^d(\tilde{b}(t) - \tilde{b}(t_0)) + \gamma_1 H_2^d(y(t) - y(t_0)) + H_1^d q_{1d}(t) + H_2^d q_{2d}(t), \quad (12)$$

where  $H_1^d = -\frac{\lambda^2}{B_1}$ ,  $H_2^d = -\frac{[\lambda^2(1 - e^{-A_2 d}) + A_2(A_2 - 2\lambda)]}{B_1 A_2 e^{-A_2 d}}$ , and

$$\begin{aligned} \begin{bmatrix} q_{1d}(t) \\ q_{2d}(t) \end{bmatrix} &= -\frac{B_1}{A_2} \frac{d}{L} \left\{ \begin{bmatrix} 0 \\ -A_2 \end{bmatrix} \left[ p(t-d) - p(t_0-d) \right] \right. \\ &\quad + \begin{bmatrix} 1 - e^{-\frac{d}{L} A_2} \\ -A_2 e^{-\frac{d}{L} A_2} \end{bmatrix} \left[ p\left(t + \frac{d}{L}(1-L)\right) \right. \\ &\quad \left. \left. - p\left(t_0 + \frac{d}{L}(1-L)\right) \right] + \dots + \right. \\ &\quad \left. \begin{bmatrix} 1 - e^{-\frac{d}{L}(L-1)A_2} \\ -A_2 e^{-\frac{d}{L}(L-1)A_2} \end{bmatrix} \left[ p\left(t - \frac{d}{L}\right) \right. \right. \\ &\quad \left. \left. - p\left(t_0 - \frac{d}{L}\right) \right] \right\}. \end{aligned}$$

Figure 2 shows that PI-type memory AQM has a slightly higher utilization (86%) than PI-type memoryless AQM (82%) in the presence of large propagation delays, while they have similar performance 92% when  $d_s = 250$ ms. Both of them have very small queuing delays and are stabilizing. From this simulation result, we can see that the stabilizing

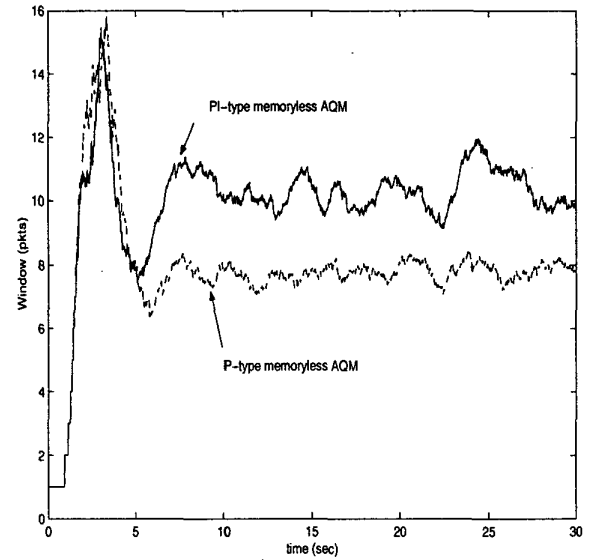


Fig. 1. Average window  $w(t)$  trajectory

AQMs based on virtual queue dynamics are not so sensitive against large delays, while the stabilizing memoryless AQMs based on real queue dynamics are as shown in *ns* simulation of [17], [18], [19]. This difference seems to come from the following fact: our previous works in [17], [18], [19] assume that the forward delay from source to router, which produces the state-delay, is zero, while the current work removes the state-delay by making the queuing delay zero with the virtual queue dynamics. Thus, the memoryless AQM based on real queue dynamics is much easier to be fragile in the presence of large delays than that based on virtual queue dynamics, i.e., the AQM based on real queue dynamics needs to compensate for large delays explicitly as suggested in [17], [18], [19].

## V. CONCLUSION

In this paper, we studied how to design a stabilizing AQM in the presence of arbitrary delays as a series of our previous works [17], [18], [19]. Our study based on a modified virtual queue dynamics shows that we can compensate for the delays explicitly by making the equilibrium queuing delay zero and as a subsidiary result, that the AVQ is a natural state-feedback control for TCP Reno based on the virtual queue dynamics. Our *ns* simulation study demonstrates an interesting phenomenon that the memoryless AQM based on virtual queue dynamics is robust against large delays, while memoryless AQMs based on the real queue dynamics in [17], [18], [19] are not due to the assumption that the forward delay from source to router is zero.

We hope that this work is helpful to design a stabilizing feedback control in the presence of arbitrary delays for future networks.

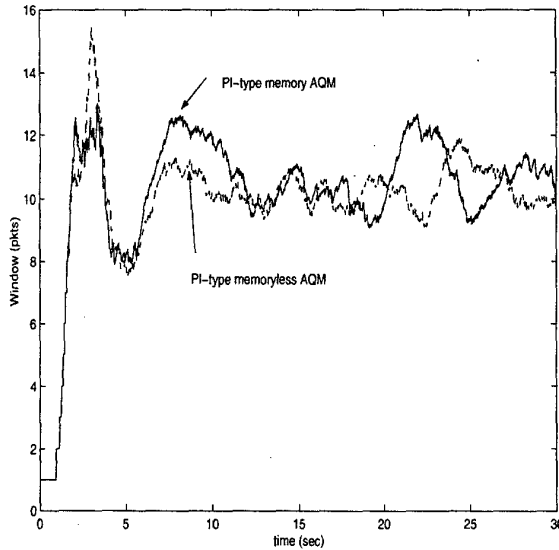


Fig. 2. Average window  $w(t)$  trajectory when  $d_s = 250\text{ms}$

## APPENDIX

### VI. DERIVATION OF (5)

Let  $(w^*, b^*, p^*)$  be the equilibrium point. Then the linearized model of TCP Reno (2) is

$$\begin{aligned} \delta \ddot{b}(t) &= \frac{\partial f}{\partial \dot{b}(t)} \Big|_* \delta \dot{b}(t) + \frac{\partial f}{\partial \dot{b}(t-d)} \Big|_* \delta \dot{b}(t-d) \\ &\quad + \frac{\partial f}{\partial p(t-d)} \Big|_* \delta p(t-d), \end{aligned}$$

where  $\frac{\partial f}{\partial \dot{b}(t)} = -\frac{N}{\gamma_2 d^2 c} (1 - p^*) - \frac{\gamma_2 p^* c}{2N}$ ,  $\frac{\partial f}{\partial \dot{b}(t-d)} = 0$ ,  $\frac{\partial f}{\partial p(t-d)} = -\frac{\gamma_1 N}{d^2} - \frac{\gamma_1 (\gamma_2 c)^2}{2N}$ , and  $p^* = \frac{2N^2}{(2N^2 + \gamma_2^2 c^2 d^2)}$ .

From this, the linearized model of TCP Reno (2) can be converted to (3).

Next, we derive (5). Differentiating (2), we can get

$$\begin{aligned} \ddot{b}(t) &= \frac{\partial f}{\partial \dot{b}(t)} \ddot{b}(t) + \frac{\partial f}{\partial \dot{b}(t-d)} \ddot{b}(t-d) \\ &\quad + \frac{\partial f}{\partial p(t-d)} \dot{p}(t-d) \\ &= g(\dot{b}(t), \ddot{b}(t), \dot{b}(t-d), \ddot{b}(t-d), \dot{p}(t-d)). \end{aligned} \quad (13)$$

The linearized model of (13), which leads to (5), can be similarly obtained as

$$\begin{aligned} \delta \ddot{b}(t) &= \frac{\partial g}{\partial \dot{b}(t)} \Big|_* \delta \dot{b}(t) + \frac{\partial g}{\partial \ddot{b}(t)} \Big|_* \delta \ddot{b}(t) + \\ &\quad \frac{\partial g}{\partial \ddot{b}(t-d)} \Big|_* \delta \ddot{b}(t-d) + \frac{\partial g}{\partial \dot{p}(t-d)} \Big|_* \delta \dot{p}(t-d) \\ &= \frac{\partial f}{\partial \dot{b}(t)} \Big|_* \delta \ddot{b}(t) + \frac{\partial f}{\partial \dot{b}(t-d)} \Big|_* \delta \ddot{b}(t-d) \\ &\quad + \frac{\partial f}{\partial p(t-d)} \Big|_* \delta \dot{p}(t-d). \end{aligned}$$

### VII. DERIVATION OF (7)

Note that system (5) can be written as

$$\begin{aligned} z(t+d) &= e^{Ad} z(t) + \int_t^{t+d} e^{A(t+d-\sigma)} B \delta \dot{p}(\sigma-d) d\sigma \\ &= e^{Ad} \left[ z(t) + \int_{-d}^0 e^{-A(\sigma+d)} B \delta \dot{p}(\sigma+t) d\sigma \right], \end{aligned}$$

where  $e^{At} = \begin{bmatrix} 1 & -\frac{1}{A_2} + \frac{e^{A_2 t}}{A_2} \\ 0 & e^{A_2 t} \end{bmatrix}$ . Define the inside part of the above second equation as  $h_e(t) = z(t) + \int_{-d}^0 e^{-A(\sigma+d)} B \delta \dot{p}(\sigma+t) d\sigma = z(t) + \begin{bmatrix} \dot{u}_{1d}(t) \\ \dot{u}_{2d}(t) \end{bmatrix}$ .

From this definition and the above derivation, system (5) can be easily rewritten as the following nominal-system:

$$\dot{h}_e(t) = A h_e(t) + \tilde{B} \delta \dot{p}(t), \quad (14)$$

where  $\tilde{B} = \begin{bmatrix} \tilde{B}_1 \\ \tilde{B}_2 \end{bmatrix} = e^{-Ad} B = \begin{bmatrix} -\frac{1}{A_2} + \frac{e^{-A_2 d}}{A_2} \\ e^{-A_2 d} \end{bmatrix} B_1$ .

To get the explicit stabilizing optimal gain as shown in Proposition 1, we have another transform. Let  $s_e(t) = T_e h_e(t)$ , where  $T_e = \begin{bmatrix} \frac{A_2 e^{-A_2 d}}{(1-e^{-A_2 d})} & 1 \\ 0 & \frac{A_2}{(1-e^{-A_2 d})} \end{bmatrix}$ . Then, (14) can be rewritten as  $\dot{s}_e(t) = T_e A T_e^{-1} s_e(t) + T_e \tilde{B} \delta \dot{p}(t)$ , which is equal to (7).

## VIII. REFERENCES

- [1] V. Jacobson, "Congestion avoidance and control," in *Proceedings of ACM/SIGCOMM*, 1988. <ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z>.
- [2] F. Baccelli and D. Hong, "AIMD, fairness and fractal scaling of TCP traffic," in *Proceedings of IEEE Infocom*, April 2002.
- [3] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, 1993. <ftp://ftp.ee.lbl.gov/papers/early.ps.gz>.
- [4] T. J. Ott, T. V. Lakshman, and L. Wong, "SRED: Stabilized RED," in *Proceedings of IEEE Infocom*, March 1999. <ftp://ftp.bellcore.com/pub/tjo/SRED.ps>.
- [5] M. Christiansen, K. Jeffay, D. Ott, and F. D. Smith, "Tuning RED for web traffic," in *Proceedings of ACM/SIGCOMM*, 2000.
- [6] W. Feng, K. G. Shin, D. Kandlur, and D. Saha, "Stochastic Fair Blue: A queue management algorithm for enforcing fairness," in *Proceedings of INFOCOM*, April 2001.
- [7] S. Kunniyur and R. Srikant, "End-to-end congestion control schemes: utility functions, random losses and ECN marks," in *Proceedings of IEEE/INFocom*, 2000. <http://www.ieee-infocom.org/2000/papers/401.ps>.
- [8] S. Athuraliya, V. H. Li, S. H. Low, and Q. Yin, "REM: active queue management," *IEEE Network*, vol. 15, no. 3, pp. 48–53, 2001. <http://netlab.caltech.edu/pub.html>.
- [9] V. Misra, W.-B. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting tcp flows with an application to RED," in *Proceedings of ACM/SIGCOMM*, September 2000.
- [10] C. Holot, V. Misra, D. Towsley, and W.-B. Gong, "A control theoretic analysis of RED," in *Proceedings of IEEE/INFocom*, April 2001. <http://www-net.cs.umass.edu/papers/papers.html>.
- [11] C. Holot, V. Misra, D. Towsley, and W. B. Gong, "On designing improved controllers for AQM routers supporting TCP flows," in *Proceedings of IEEE/INFocom*, 2001. <http://www-net.cs.umass.edu/networks/publications.html>.
- [12] S. Kunniyur and R. Srikant, "Analysis and design of an adaptive virtual queue (AVQ) algorithm for active queue management," in *Proceedings of ACM/SIGCOMM*, 2001. <http://comm.csl.uiuc.edu/~srikant/pub.html>.
- [13] H. Lim, K. J. Park, E. C. Park, and C. H. Choi, "Active queue management algorithm with a rate regulator," in *Proc. of 15th IFAC World Congress on Automatic Control*, vol. T-We-M02, (Barcelona, Spain), 2002.
- [14] E. C. Park, H. Lim, K. J. Park, and C. H. Choi, "Analysis of the virtual rate control algorithm in TCP networks," in *Globecom*, vol. Nov, (Taipei, Taiwan), 2002.
- [15] J. H. Lee, S. W. Kim, and W. H. Kwon, "Memoryless  $H_\infty$  controllers for state delayed systems," *IEEE Trans. on Autom. Contr.*, vol. 39, pp. 159 – 162, 1994.
- [16] P. Park, "A delay-dependent stability criterion for systems with uncertain time-invariant delays," *IEEE Trans. Automat. Contr.*, vol. 44, no. 4, pp. 876–877, 1999.
- [17] K. B. Kim and S. H. Low, "Design of receding horizon AQM in stabilizing TCP with multiple links and heterogeneous delays," in *Proc. of 4th Asian Control Conference*, vol. WA-1, (Singapore), 2002. <http://cisl.snu.ac.kr/~kbb>.
- [18] K. B. Kim and S. H. Low, "Analysis and design of AQM based on state-space models for stabilizing TCP," in *Proc. of IEEE American Control Conference*, (Denver, Colorado), pp. 260–265, 2003.
- [19] K. B. Kim, A. Tang, and S. H. Low, "Design of AQM in supporting TCP based on the well-known AIMD model," in *Proc. of IEEE Globecom*, vol. Dec., (San Francisco), 2003.
- [20] L. Guo and I. Matta, "The war between mice and elephants," in *Proceedings of International Conf. on Network Protocols (ICNP)*, 2001.
- [21] K. B. Kim and W. H. Kwon, "Intervalwise receding horizon  $H_\infty$  tracking controls for linear continuous time-varying systems," in *Proc. of IEEE American Control Conference*, (Anchorage, Alaska, USA), pp. 2186–2191, May, 2002.
- [22] H. K. Khalil, *Nonlinear systems*. Prentice Hall, 1996.